538, 100

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
24 June 2004 (24.06.2004)

PCT

(10) International Publication Number
WO 2004/054183 A1

(51) International Patent Classification[7]: H04L 12/56

(21) International Application Number:
PCT/IB2003/005879

(22) International Filing Date:
10 December 2003 (10.12.2003)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/432,893 12 December 2002 (12.12.2002) US

(71) Applicant *(for all designated States except US)*: **KONINKLIJKE PHILIPS ELECTRONICS N.V.** [NL/NL]; Groenewoudseweg 1, NL-5621 BA Eindhoven (NL).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **LI, Qiong** [US/US]; P.O. Box 3001, Briarcliff Manor, NY 10510-8001 (US). **VAN DER SCHAAR, Mihaela** [US/US]; P.O. Box 3001, Briarcliff Manor, NY 10510-8001 (US).

(74) Common Representative: **KONINKLIJKE PHILIPS ELECTRONICS N.V.**; c/o WAXLER, Aaron, P.O. Box 3001, Briarcliff Manor, NY 10510-8001 (US).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

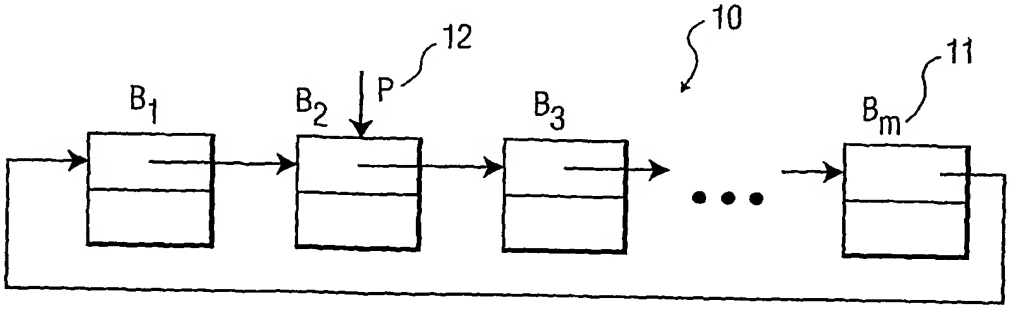
(84) Designated States *(regional)*: ARIPO patent (BW, GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PT, RO, SE, SI, SK, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Declaration under Rule 4.17:
— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii)) for all designations*

Published:
— *with international search report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: A SYSTEM AND METHOD USING A CIRCULAR BUFFER FOR DETECTING PACKET LOSS IN STREAMING APPLICATIONS



(57) Abstract: A circular buffer, i.e., a chain of buffers forming a circle, is provided for managing packet loss detecting in Internet streaming. The detection latency is determined by the size of the buffer chain, which can be dynamically adapted to network conditions and application requirements. The present invention can achieve reasonable detection accuracy.

WO 2004/054183 A1

A SYSTEM AND METHOD USING A CIRCULAR BUFFER FOR DETECTING PACKET LOSS IN STREAMING APPLICATIONS

This invention relates to the detection of packet loss in Internet streaming. More particularly, this invention relates to a system and method for using a circular buffer, implemented by a chain of buffers forming a circle, for packet loss detection in Internet streaming. Most particularly, this invention relates to a system and method for dynamically adapting packet loss detection latency, which is determined by the size of the chain, to network conditions and application requirements that achieves reasonable detection accuracy and is easy to implement.

In Internet streaming applications, when an important packet gets lost, such as a packet belonging to an I-frame in video or the base-layer in scalable coding, the receiver may ask the sender to retransmit this lost packet. In order to send a retransmission request promptly, the receiver must have means to timely detect packet losses. Currently, packet loss detection is done using either a timer or timing windows.

In the transmission control (TCP) protocol, a timer is used for loss detection. When a packet is sent, a timer with a timeout value is set by the sender for that packet. If the timer expires before the acknowledgement of the packet is received, the packet is declared as lost and resent by the sender.

In most streaming applications, loss detection is done by using timing windows. A timing window (or more precisely a table) has a fixed number of binary entries. Each entry indicates a packet 's status (0: lost, 1: received). At a certain point of time, the first entry in this window is associated with a packet that is identified by a sequence number (such as the sequence number of a real-time transport protocol (RTP) packet). The subsequent window entries are associated with packets having higher sequence numbers in sequential order. Therefore, the space of packet sequence numbers can be viewed as being divided into blocks, each block being associated with a window at a certain point of time.

Presently, two such timing windows associated with two consecutive blocks of packets are used for packet loss detection. In the beginning, all entries are marked as "0". When a packet is received, the corresponding window entry is marked as "1". When a packet is received that has a sequence number that goes beyond the first window, the corresponding second window entry is marked. As soon as a packet is transmitted with a sequence number that goes beyond even the second window, the first window is closed, and its entries are checked. Packets associated with entries that remain marked as "0", are declared as lost. A consecutive new window is opened right after the second window and the detection process resumes.

The timer method can be applied only to TCP-like protocols that can measure packet round-

trip time in order to properly set the timeout value of the timer. In streaming applications, most of the time only unidirectional media streams are generated. The timer method is not applicable to these cases. Instead, timing window methods are used. However, there are limitations with the timing window methods:

- The window size is fixed - there is no built-in mechanism for window size adaptation, which is desirable when network conditions and application requirements (e.g. delay) change.
- Non-uniform loss detection latency for different losses occurs - the lost detection latency lies in a range between T and ~ 2T, where T is the average period of the timing window, such that when a loss is associated with the first entry of the window, the detection latency is 2T, while it is T if the loss is associated with the last entry of the window.

Thus, there is a need for a loss detection method that allows adaptive loss detection latency, as well as a uniform loss detection latency. The system and method of the present invention comprises:

- a circular chain of buffers having a chain size that is adjustable according to network conditions and application requirements, thereby providing adaptive loss detection latency; and
- a circular chain of buffers having a fixed chain size, thereby providing uniform detection latency when the chain size is fixed.

Shortening the latency can increase the chance of recovering a lost packet. Having a uniform latency may imply an equal recovery chance for all losses. Therefore having an equal latency for all loss detection may be desirable for many streaming applications.

The implementation overhead of the circular buffer of the present invention is low and can be less than the timing window method.

FIG. 1a illustrates a preferred embodiment of the circular buffer structure of the present invention.

FIG. 1b illustrates the structure of each of the buffers in the circular buffer illustrated in FIG. 1a.

FIG. 2a illustrates an algorithm of a preferred that implements the circular buffer structure illustrated in FIGs. 1a-b.

FIG. 2b illustrates a flow chart of the algorithm illustrated in FIG. 2a.

FIG. 3 illustrates a preferred embodiment of an algorithm for adapting the structure of the circular buffer chain based on network characteristics.

Referring now to FIB. 1a, buffers 1 through m form a circularly linked list of m buffers B_i 10 where $i= 1, ..., m$ each of whose structure 12 is shown in FIG. 1b. As shown in FIG. 1a, m is the length of the circular chain that determines the loss detection latency, and it can be adapted to network conditions and application requirements. P 11 is a pointer that circulates through the chain, pointing to each buffer in turn. Each buffer B_i 10 in the chain comprises two fields, F_1 13 and F_2 14. F_1 13 stores a pointer to the next buffer. F_2 14 stores a sequence number s of a packet that may get lost.

In streaming applications, packets are supposed sent in the order of packet sequence number. In a no-loss and ideal world, an arriving packet always has a sequence number one higher than the previous. If a packet arrives out of order, or is lost, then a hole or gap is observed in the sequence numbers of the received packets. Whenever a hole (could be a hole that spans more than one consecutive number) is observed, potentially, this hole may indicate one or more lost packets. However, out-of-order packet delivery is common to Internet because each packet can take a different path through the network and an earlier numbered packet may take longer to arrive than

a later numbered packet. An application cannot make a loss declaration immediately after observing a hole in the sequence of arriving packets. The application has to wait and see whether this hole is just an incident of out-of-order delivery. The circular buffer method provides a way to determine if a loss declaration can be made.

FIG. 2a illustrates C programming language code for a preferred embodiment of an algorithm for accomplishing the method of the present invention. FIG. 2b is a flow chart of the algorithm illustrated in FIG. 2a. The pointer P 12 circulates through the chain of buffers B_i 11, the circulation being driven by receipt of a packet at step 20. The sequence number of the received packet is checked against that of the current maximum sequence number already received s at step 21, and if it is less than the current maximum sequence number received it is an out of order packet.

If it is not an out of order packet, at step 22 a hole in the sequence is checked for and when a hole in the sequence is observed the following steps are performed:

a. If the buffer at which P is pointing contains a non-received packet ($P->F2$ is not zero at step 24) then at step 25 the packet with the sequence number $P->F2$ is declared lost.

b. Then, regardless of whether or not P was pointing at a non-received packet, at step 26 the current maximum sequence number is incremented by one and stored in the current buffer and P is updated to point to the next buffer in sequence, i.e., $P=P->F1$.

c. The number of buffers that fall in the hole is decremented by 1 at step 27 and steps a-c are repeated until the remaining number of buffers is zero.

Thus, all the sequence numbers that fall in the hole are stored in the circular buffer chain, with each number occupying one buffer.

When a hole is not observed, the following steps are performed:

d. At step 28 if the buffer at which P is pointing contains a non-received packet (P->$F2$ is not zero) then at step 29 the packet with the sequence number P->$F2$ is declared lost and the sequence number stored in the buffer is set to zero

e. Whether or not P->$F2$ points at a non-received packet, at step 30 P is updated to point to the next buffer in sequence.

When all the processing associated with an in order packet is completed:

f. At step 31 the current maximum sequence number is set to that of the received packet.

If a packet arrives out of order (having a sequence number that is earlier than the current received maximum sequence number s):

g. the received packet number is compared with the numbers stored in the circular buffer, and the corresponding record in the buffer is cleaned, i.e., set to zero at step 32.

Thus, the detection latency is determined by the size of the buffer chain m, because the loss declaration is only made when the pointer re-visits a non-empty buffer, i.e., when F_2 is non-zero.

As illustrated in FIG. 3, the chain size m, that determines the detection latency can be adapted. For example, in a preferred embodiment, initially $m = 4$. If the observed false declaration rate is higher than a given threshold, i.e.,

$$false\_rate > TOLERABLE\_RATE$$

the length may be too short and may need to be lengthened by inserting a new buffer and adjusting m correspondingly 36. The greater the length of the network path, i.e., number of links traversed, the larger the m that is needed, because when a packet

traverses a longer network path, there is a greater likelihood of out-of-order delivery occurring. The larger value of m decreases the likelihood of the pointer P encountering a delivered but out-of-order packet in a buffer as P circulates through the buffer chain B_i 11.

The *success_rate* is initially declared to be a pre-determined *EXPECT_RATE* and adjusted thereafter to be

$$success\_rate = \frac{declared\_losses - falsely\_declared\ losses}{declared\_losses}$$

$$false\_rate = 1 - success\_rate$$

and if the *success_rate* is too high, i.e., if

$$success\_rate > EXPECT\_RATE$$

the length of the buffer chain may be too long and may need to be shortened by deleting a buffer as illustrated in FIG. 3.

This present invention can be used in the implementation of multimedia players that play media from networked storage. Or it can be used by any type of multimedia receiver that wants to use retransmission as an error-recovery means, therefore need to perform packet loss detection. Finally, it can be used by transport control protocol implementations that the packet loss detection is done at the receiver side.

The methods and systems of the present invention, as described above and shown in the drawings, provide for a circular buffer that allows an adaptive latency detection time or a fixed latency detection time. It will be apparent to those skilled in the art that various modifications and variations can be made in the method and system of the present invention without departing from the spirit or scope of the invention.

Thus, it is intended that the present invention includes modifications and variations that are within the scope of the appended claims and their equivalents.

CLAIMS:

1. A system for adaptive detection of streamed packet loss by a receiver of a plurality of streamed packets transmitted over a network from a given sender to the receiver, comprising:

a circular buffer (10) of size $m>1$ entries (13) , each entry (13) having at most one sequence number (15) of a streamed packet that has not been received and is possibly lost;

a packet loss detection module (33) that uses the circular buffer (10) to detect and store therein a sequence number (15) of a non-received and possibly lost packet, to detect therein and remove therefrom a sequence number (15) of a lost packet and declare the packet lost, and to remove therefrom a sequence number (15) of a possibly lost packet that is received from the given sender;

an adaptation module (37) that adapts the system to a network condition,

wherein a loss detection latency is determined by the size m of the circular buffer (10) and the loss declaration is possibly false.

2. The system of claim 1, wherein m is initially set to 4.

3. The system of claim 1, further comprising:

a variable s having an initial value of 1 and being adapted to store a highest sequence number of a streamed packet transmitted over the network from the given sender and received by the receiver;

a pointer *P (12)* having an initial position pointing at a pre-determined location in the circular chain and being adapted to circulate sequentially through the *m* entries of said circular buffer (10) beginning at an entry (13) in the circular buffer (10) that is next in sequence to the entry (13) corresponding to the variable *s*;

wherein,

for a streamed packet received from the given sender, the packet loss detection module (33) checks the sequence number of the received packet against the variable *s* and performs one of the following -

a. if a hole in the sequence of received packets is observed beginning at the location pointed at by the pointer *P (12)*, each entry (13) of the circular buffer (10) that is in the hole is checked for a sequence number (15) of a possibly lost packet and the corresponding packet is declared lost, a total of declared losses *declared_losses* is increased by one, each sequence number in the hole is stored in ascending order in a sequential entry (13) beginning at the location pointed at by the pointer *P (12)*, *P (12)* is updated to point to the entry (13) in the circular buffer (10) following hole, and *s* is set equal to the sequence number of the received pkt,

b. if a hole in the sequence of received packets in not observed the entry (13) pointed at by the pointer *P (12)* is checked for a sequence number (15) of a possibly lost packet and the corresponding packet is declared lost, a total of declared losses *declared_losses* is increased by one, the entry (13) is cleared, *P (12)* is updated to point to the next entry (14) in the circular buffer (10), and *s* is set equal to the sequence number of the received pkt,

c. if an out of order packet is observed, the entries of the circular buffer (10) are searched to find one that contains a sequence number (15) equal to the sequence

number of the received packet and if found the entry (13) is cleared, if not found a false declaration rate *false_declared_losses* is increased by one.

4. The system of claim 3, wherein:

the network condition is at least one of a success rate of transmission (*success_rate)* and the false declaration rate (*false_rate)* wherein the *success_rate* is initially set to a pre-determined expected rate (*EXPECT_RATE)*; and

the adaptation module (37) adjusts the size *m* of the circular buffer (10) according to the network condition as follows

a. *m* is increased if *false_rate* > *TOLERABLE_RATE* where *TOLERABLE_RATE* is a predetermined threshold and an entry (13) is added to the circular buffer (10), or

b. *m* is decreased if

$$success\_rate = \frac{declared\_losses - falsely\_declared\_losses}{EXPECT\_RATE} > EXPECT\_RATE$$

and an entry (13) is removed from the circular buffer (10).

5. The system of claim 4, wherein the circular buffer (10) is a circular buffer (10) chain

Bi for *i=1, ..., m* of a plurality of *m>1* buffers such that each of said plurality of buffers is an entry (13) comprising a pointer to the next buffer (14) in the chain and a value for storing a sequence number (15) of a non-received buffer and the pointer *P (12)* point to a buffer in the chain.

6. A system for adaptive detection of streamed packet loss by a receiver of a plurality of streamed packets transmitted over a network from a given sender to the receiver, comprising:

a circular buffer (10) of size $m>1$ entries, each entry (13) having at most one sequence number (15) of a streamed packet that has not been received and is possibly lost;

a packet loss detection module (33) that uses the circular buffer (10) to detect and store therein a sequence number (15) of a non-received and possibly lost packet, to detect therein and remove therefrom a sequence number (15) of a lost packet and declare the packet lost, and to remove therefrom a sequence number (15) of a possibly lost packet that is received from the given sender;

means for adapting the system to a network condition (37),

wherein a loss detection latency is determined by the size m of the circular buffer (10) and the loss declaration is possibly false.

7. The system of claim 6, further comprising:

a variable s having an initial value of 1 and being adapted to store a highest sequence number of a streamed packet transmitted over the network from the given sender and received by the receiver;

a pointer *P (12)* having an initial position pointing at a pre-determined location in the circular chain and being adapted to circulate sequentially through the m entries of said circular buffer (10) beginning at an entry (13) in the circular buffer (10) that is next in sequence to the entry (13) corresponding to the variable s;

wherein,

for a streamed packet received from the given sender, the packet loss detection module (33) checks the sequence number of the received packet against the variable *s* and performs one of the following -

a. if a hole in the sequence of received packets is observed beginning at the location pointed at by the pointer *P (12)*, each entry (13) of the circular buffer (10) that is in the hole is checked for a sequence number (15) of a possibly lost packet and the corresponding packet is declared lost, a total of declared losses *declared_losses* is increased by one, each sequence number in the hole is stored in ascending order in a sequential entry (13) beginning at the location pointed at by the pointer *P (12)*, *P (12)* is updated to point to the entry (13) in the circular buffer (10) following hole, and *s* is set equal to the sequence number of the received pkt,

b. if a hole in the sequence of received packets in not observed the entry (13) pointed at by the pointer *P (12)* is checked for a sequence number (15) of a possibly lost packet and the corresponding packet is declared lost, a total of declared losses *declared_losses* is increased by one, the entry (13) is cleared, *P (12)* is updated to point to the next entry (14) in the circular buffer (10), and *s* is set equal to the sequence number of the received pkt,

c. if an out of order packet is observed, the entries (13) of the circular buffer (10) are searched to find one that contains a sequence number (15) equal to the sequence number of the received packet and if found the entry (13) is cleared, if not found a false declaration rate *false_declared_losses* is increased by one.

8. A method for adaptive detection of streamed packet loss by a receiver of a plurality of streamed packets transmitted over a network from a given sender to the receiver, comprising the steps of:

providing a circular buffer (10) of size $m>1$ entries, each entry (13) having at most one sequence number (15) of a streamed packet that has not been received and is possibly lost;

receiving from the given sender a streamed packet having a sequence number;

using the circular buffer (10) and the sequence number of the received packed to perform one of the steps of:

a. detecting and storing in the circular buffer (10) a sequence number (15) of a non-received and possibly lost packet,

b. detecting in the circular buffer (10) and removing therefrom a sequence number (15) of a lost packet and declaring the packet lost such that the loss declaration is possibly false, and

c. removing from the circular buffer (10) a sequence number (15) of a possibly lost packet that corresponds to the sequence number of the received packet;

adapting the method to a network condition such that a loss detection latency is determined by the size m of the circular buffer (10).

9. The method of claim 8, further comprising the steps of:

providing a variable s having an initial value of 1;

setting the provided variable s = max(s, sequence number of the received streamed packet);

providing a pointer *P (12)* having an initial position pointing at a pre-determined location in the circular buffer (10) that is adapted to circulate sequentially through the *m* entries of the provided circular buffer (10) beginning at an entry (13) in the circular buffer (10) that is next in sequence to the entry (13) corresponding to the variable *s*;

for a streamed packet received from the given sender, checking the sequence number of the received packet against the variable *s* and performing one of the following steps-

a. if a hole in the sequence of received packets is observed beginning at the location pointed at by the pointer *P (12)*,

a.1 checking each entry (13) of the circular buffer (10) that is in the hole for a sequence number (15) of a possibly lost packet and declaring the corresponding packet lost,

a.2 if a packet is declared lost, increasing a total of declared losses *declared_losses* by one,

a.3 storing each sequence number in the hole in ascending order in a sequential entry (13) beginning at the location pointed at by the pointer *P (12)*,

a.4 updating the pointer *P (12)* to point to the entry (13) in the circular buffer (10) following hole, and

a.5 setting *s* equal to the sequence number of the received pkt;

b. if a hole in the sequence of received packets in not observed

b.1 checking the entry (13) pointed at by the pointer *P (12)* for a sequence number (15) of a possibly lost packet and declaring the corresponding packet lost,

b2. if a packet is declared lost, increasing a total declared losses *declared_losses* by one,

b.3 clearing the entry (13) pointed at by *P (12)*,

b.4 updating *P (12)* to point to the next entry (14) in the circular buffer (10), and

b.5 setting *s* equal to the sequence number of the received pkt;

c. if an out of order packet is observed

c.1 searching the entries (13) of the circular buffer (10) to find one that contains a sequence number (15) equal to the sequence number of the received packet,

c.2 if found, clearing the entry (13),

c.3 if not found, increasing a false declaration rate *falsely_declared_losses* by one.

10. The method of claim 9, wherein:

the network condition is at least one of a success rate of transmission (*success_rate)* and the false declaration rate (*false_rate))* wherein the *success_rate* is initially set to a pre-determined expected rate (*EXPECT_RATE)*; and

the adaptation step adjusts the size *m* of the circular buffer (10) according to the network condition by performing one of the following steps:

d. if *false_rate* > *TOLERABLE_RATE* where *TOLERABLE_RATE* is a predetermined

threshold performing the following steps

d.1 increasing *m* by 1, and

d.2 adding an entry (13) to the circular buffer (10), or

e. if *success_rate* > *EXPECT_RATE*

e.1 decreasing *m* by 1,

e.2 removing an entry (13) from the circular buffer (10),

11. The method of claim 9, wherein *m* is initially set to 4.

12. A computer program product for use in conjunction with a processor to adapt detection of streamed packet loss by a receiver of a plurality of streamed packets transmitted over a network from a given sender to the receiver, the computer program product comprising a computer readable storage medium and a computer program mechanism embedded therein, the computer program mechanism comprising:

a circular buffer (10) of size *m>1* entries, each entry (13) having at most one sequence number (15) of a streamed packet that has not been received and is possibly lost;

a packet loss detection routine (33) including instructions for using the circular buffer (10) to detect and store therein a sequence number (15) of a non-received and possibly lost packet, detect therein and remove therefrom a sequence number (15) of a lost packet and declare the packet lost, and remove therefrom a sequence number (15) of a possibly lost packet that is received from the given sender;

an adaptation routine (37) including instructions that adapt the system to a network condition,

wherein a loss detection latency is determined by the size *m* of the circular buffer (10) and the loss declaration is possibly false.

13. The computer program product of claim12, further comprising:

a variable *s* having an initial value of 1 and being adapted to store a highest sequence number of a streamed packet transmitted over the network from the given sender and received by the receiver;

a pointer *P (12)* having an initial position pointing at a pre-determined location in the circular chain and being adapted to circulate sequentially through the *m* entries of said circular buffer (10) beginning at an entry (13) in the circular buffer (10) that is next in sequence to the entry (13) corresponding to the variable *s*;

wherein,

for a streamed packet received from the given sender, the instructions of the packet loss detection routine (33) check the sequence number of the received packet against the variable *s* and perform one of the following -

a. if a hole in the sequence of received packets is observed beginning at the location pointed at by the pointer *P (12)*, each entry (13) of the circular buffer (10) that is in the hole is checked for a sequence number (15) of a possibly lost packet and the corresponding packet is declared lost, total of declared losses *declared_losses* is increased by one, each sequence number in the hole is stored in ascending order in a sequential entry (13) beginning at the location pointed at by the pointer *P (12)*, *P (12)* is updated to point to the entry (13) in the circular buffer (10) following hole, and *s* is set equal to the sequence number of the received pkt,

b. if a hole in the sequence of received packets in not observed the entry (13) pointed at by the pointer *P (12)* is checked for a sequence number (15) of a possibly lost packet and the corresponding packet is declared lost, a total of declared

losses *declared_losses* is increased by one, the entry (13) is cleared, *P (12)* is updated to point to the next entry (14) in the circular buffer (10), and *s* is set equal to the sequence number of the received pkt,

c. if an out of order packet is observed, the entries (13) of the circular buffer (10) are searched to find one that contains a sequence number (15) equal to the sequence number of the received packet and if found the entry (13) is cleared, if not found a false declaration rate *falsely_declared_losses* is increased by one and the total of *declared losses declared_losses* is decreased by one.

14. The computer program product of claim 13, wherein:

the network condition is at least one of a success rate of transmission *(success_rate)* and a false declaration rate *(false)rate)* wherein the *success_rate* is initially set to a pre-determined expected rate *(EXPECT_RATE)*; and

the adaptation routine (37) adjusts the size *m* of the circular buffer (10) according to the network condition as follows

a. *m* is increased if *false_rate* > *TOLERABLE_RATE* where *TOLERABLE_RATE* is a predetermined threshold and an entry (13) is added to the circular buffer (10), or

b. *m* is decreased if *success_rate* = $\frac{declared\_losses - falsely\_declared\_losses}{EXPECT\_RATE} > EXPECT\_RATE$ and an entry (13) is removed from the circular buffer (10).

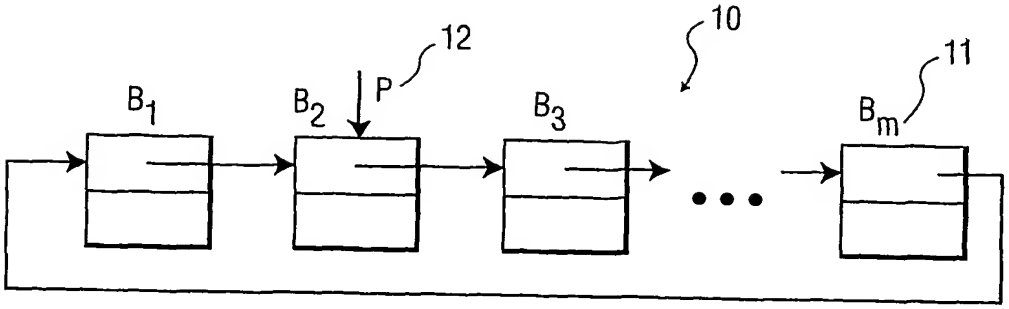


FIG. 1A

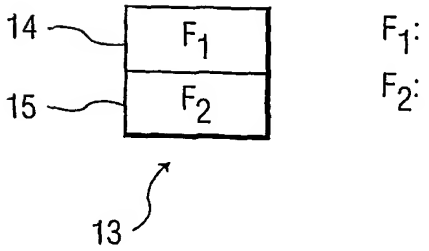


FIG. 1B

CIRCULAR BUFFER INITIALIZATION:

$B_i \rightarrow F_2 = 0$, FOR i = 1,2,...m
$P = B_1$

NORMAL OPERATION:
VARIABLE:

s: CURRENT PACKET SEQUENCE NUMBER
h: LENGTH OF OBSERVED HOLE IN SEQUENCE NUMBER
P: THE POINTER THAT CIRCULATES THE CIRCULAR BUFFER

```
UPON RECEIVING A PACKET WITH SEQ. NO. x
{
        h=x-s;
        if (h>0)
        {
                if (h=1)  //NO HOLE
                {
                        if (P→F2 ≠ 0) {
                                declare the packet with the seq. No of P→F2 lost;
                                P→F2 = 0;
                        }
                        P= P→F1; //MOVE THE POINTER TO NEXT BUFFER
                }

                while (h >1)
                {

                        if (h>1) //HAS HOLE
                        {
                                if (P→F2 ≠ 0) {
                                    declare the packet with the seq. No of P→F2 lost;
                                    }
                                P→F2 = ++s; //A POSSIBLE LOSS
                                P = P→F1; //MOVE THE POINTER TO NEXT BUFFER
                        }
                        h--;
                }
                s=x; //UPDATE THE CURRENT SEQUENCE NUMBER
        } else //RECEIVE AN OUT OF ORDER PACKET
        {
                find out Bi that Bi→F2 =x, do
                        Bi→F2 =0; //CLEAN THE RECORD.
        }
}
```

FIG. 2A

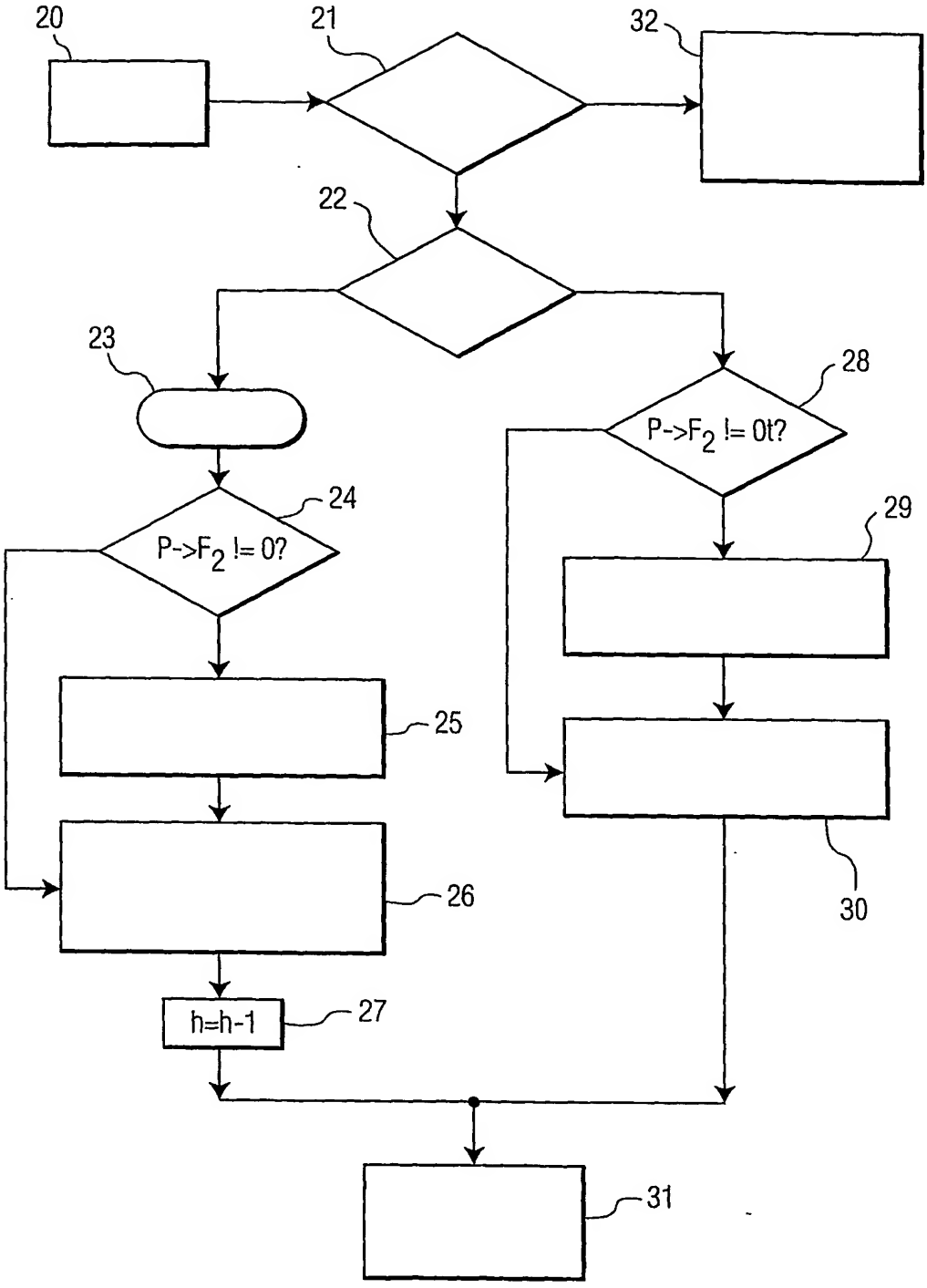


FIG. 2B

chain_size adaptation:

INITIALIZE:

```
m=4; //INITIAL CHAIN SIZE
success_rate=EXPECT_RATE;   //DECLARED SUCCESS RATE
            //DEFINED AS THE RATIO OF TRUE LOSSES TO TOTAL DECLARED LOSSES
false_rate=0;        //FALSE DECLARATION RATE
                     //DEFINED AS THE RATIO OF FALSELY DECLARED LOSSES TO TOTAL DECLARED LOSSES

UPON RECEIVING A PACKET OR MAKING A LOSS DECLARATION
{
    update success_rate and false_rate;
    if MIN_CHAIN_SIZE < m < MAX_CHAIN_SIZE
    if success_rate > EXPECT_RATE
                    consider deleting a buffer from the chain and m--;
            if false_rate > TOLERABLE_RATE
```

35

36

FIG. 3

INTERNATIONAL SEARCH REPORT

PCT/IB 05879

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 H04L12/56

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal, WPI Data, INSPEC

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 5 151 899 A (HARVEY GEORGE A ET AL) 29 September 1992 (1992-09-29) column 4, line 55 - line 62 column 5, line 22 - line 33 column 6, line 7 - line 21	1,6,8,12
A	US 6 141 324 A (DEY ABHIJIT ET AL) 31 October 2000 (2000-10-31) column 14, line 50 -column 15, line 10; figure 8	1-14
A	WO 02 065704 A (BLAIR CHRISTOPHER DOUGLAS ;EYRETEL PLC (GB)) 22 August 2002 (2002-08-22) page 26, line 20 - line 34 page 18, line 19 - line 34	1-14

☐ Further documents are listed in the continuation of box C.

☒ Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
30 March 2004	06/04/2004

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Gregori, S

INTERNATIONAL SEARCH REPORT

PCT/IB[redacted]05879

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 5151899 A	29-09-1992	DE 69216704 D1	27-02-1997
		DE 69216704 T2	14-08-1997
		EP 0525174 A1	03-02-1993
		JP 5502362 T	22-04-1993
		WO 9214327 A1	20-08-1992
US 6141324 A	31-10-2000	AU 5800599 A	21-03-2000
		CA 2340679 A1	09-03-2000
		EP 1110341 A1	27-06-2001
		JP 2002524915 T	06-08-2002
		WO 0013357 A1	09-03-2000
WO 02065704 A	22-08-2002	EP 1360799 A1	12-11-2003
		WO 02065704 A1	22-08-2002